The simplest examples where the simplex method cycles and conditions where EXPAND fails to prevent cycling [1]

J.A.J. Hall and K.I.M. McKinnon [2]

Dept. of Mathematics and Statistics, University of Edinburgh, EH9 3JZ, UK

This paper introduces a class of linear programming examples which cause the simplex method to cycle indefinitely and which are the simplest possible examples showing this behaviour. The structure of examples from this class repeats after two iterations. Cycling is shown to occur for both the most negative reduced cost and steepest edge column selection criteria. In addition it is shown that the EXPAND anti-cycling procedure of Gill *et al.* is not guaranteed to prevent cycling.

Key words: Linear programming, simplex method, degeneracy, cycling, EXPAND

1 Introduction

Degeneracy in linear programming is of both theoretical and practical importance. It occurs whenever one or more of the basic variables is at its bound. An iteration of the simplex method may then fail to improve the objective function. The simple proof of finiteness of the simplex algorithm relies on a strict improvement in the objective function at each iteration and the fact that the simplex method visits only basic solutions, of which there is a finite number. However if the problem is degenerate there is the possibility of a consecutive sequence of iterations occurring with no change in the objective function and with the eventual return to a previously encountered basis. Examples such as Beale's [2] have been constructed to show that this can happen, though such examples do seem to be very rare in practice. A more common practical situation is where a long but finite sequence of iterations occurs without

[1] Work supported by EPSRC grant GR/J0842
[2] jajhall@maths.ed.ac.uk,ken@maths.ed.ac.uk

the objective function improving—a situation called *stalling*—and this can degrade the algorithm's performance.

A related issue is the behaviour of the simplex algorithm in the presence of roundoff error. At a degenerate vertex there is a serious danger of selecting pivots that are small and have a high relative error.

A wide range of methods have been suggested to avoid these problems.

Lexicographic ordering: These methods are guaranteed to terminate in exact arithmetic but are often prohibitively expensive to implement for the revised simplex method and do not address the problem of inexact arithmetic.

Primal-dual alternation: These methods were introduced by Balinski and Gomory [1] and have recently been developed by Fletcher [3,5,4]. Some of these methods guarantee to terminate in exact arithmetic and also exhibit good behaviour with inexact arithmetic.

Constraint perturbation and feasible set enlargement: These methods attempt to reduce the likelyhood of cycling and also attempt to improve the numerical behaviour and reduce the number of iterations. The Devex and EXPAND procedures described below are of this type. In addition it is claimed that stalling cannot occur with EXPAND with exact arithmetic. Wolfe's method is a recursive perturbation method which guarantees termination in exact arithmetic.

In [10] Wolfe introduced a perturbation method which is guaranteed to terminate in a finite number of steps in exact arithmetic. In this method, whenever a degenerate vertex is encountered, the bounds producing the degeneracy are expanded in such a way that the current vertex is no longer degenerate. Other bounds on the basic variables are temporarily ignored. The simplex method works on this modified problem until an unbounded direction is found. If the bound expansion is random, it is highly unlikely that further degenerate vertices will be encountered before the unbounded direction is found. However if a further degenerate vertex is discovered, it is guaranteed to have fewer active constraints. The perturbation process is repeated and after a finite number of steps a non-degenerate vertex is reached with an unbounded direction. This direction is then used in the original problem to give an edge leading out of the degenerate vertex. It is not obvious how to extend this method to the case of inexact arithmetic, as there is then no obvious criterion for what constitutes a degenerate vertex.

In [9] Harris introduced the Devex row selection method, which allowed small violations of the constraints and used the resulting flexibility to choose the largest pivot. This has the advantage of both avoiding unnecessarily small pivots and reducing the number of iterations. The disadvantage is that the

constraints are violated and some steps are negative. The variable leaving the basis does not normally do so at one of its bounds, but is shifted to that value, resulting in inconsistent values for the basic variables. The method attempts to correct this inconsistency at regular intervals (usually after each reinversion) by doing a *reset*, in which the basic variable values are recalculated from the values of the nonbasic variables. This can produce infeasible values for the basic variables (i.e. outside the specified tolerance) so there is no guarantee that progress has been made. However the method seems to be effective in practice in reducing the number of iterations taken, and variants of it are used in some commercial codes.

Gill *et al* [7] developed the EXPAND method in an attempt to improve on the good features of the Devex method of Harris and also to incorporate some features of Wolfe's method which guarantee finite termination. The performance of MINOS was significantly improved by the incorporation of EXPAND. At each iteration of the EXPAND method the bounds are expanded by a small amount. As in Devex, the largest pivot that does not lead to any constraint violation (beyond the current expanded position) is chosen. If the normal step for the largest pivot is sufficiently positive, it is taken; otherwise a small positive step is taken. In all cases the variable values stay within their expanded bounds. Because at every iteration the nonbasic variable is moved a positive amount in the direction that improves the objective function, the objective can never return to a previous value so no previous solution can recur.

In this paper we introduce and analyse the simplest possible class of cycling examples, the 2/6-cycle class. In Section 2 we present an example of this class which cycles when using the most negative reduced cost column selection criterion. In Section 3 the general form of such examples is derived. In Section 4 a variation of the example is introduced which cycles for the steepest-edge column selection rule. In Section 5 the behaviour of the EXPAND procedure is analysed and a simple necessary and sufficient condition is derived for indefinite cycling to occur.

2 Introductory example

We first solve the four variable, two constraint problem (1) by the simplex method. The analysis later in the paper shows how to derive examples of this form. The problem is unbounded. A bounded example with identical behaviour can be obtained by adding the upper bound constraints $x_1 \leq 1$ and $x_2 \leq 1$, either as implicit upper bounds or with one or more explicit constraints. The variable to enter the basis will be chosen by the most negative reduced cost criterion and, where there is a tie for the variable to leave the basis, the variable in the row with the largest pivot will be chosen.

$$
\begin{array}{ll}
\text{Max} & z = 2.3x_1 + 2.15x_2 - 13.55x_3 - 0.4x_4, \\
\text{subject to} & \quad 0.4x_1 + 0.2x_2 - 1.4x_3 - 0.2x_4 \leq 0, \\
& -7.8x_1 - 1.4x_2 + 7.8x_3 + 0.4x_4 \leq 0, \\
& \qquad\qquad\qquad x_j \geq 0, \quad j = 1 \ldots 4.
\end{array} \tag{1}
$$

After introducing slack variables x_5 and x_6 and writing the equations in detached coefficient form we get tableau $T^{(1)}$. All the variables are initially zero and will remain zero at every iteration. In the first iteration x_1 is chosen to enter the basis. There is only one positive entry in the x_1 column, so there is a unique pivot choice with x_5 leaving the basis. This basis change leads to tableau $T^{(2)}$. In the second iteration x_2 is chosen to enter the basis. In the normal ratio test there is a tie between x_6 and x_1 to leave the basis. Breaking the tie by using the larger pivot (as is normal for numerical stability) gives x_6 to leave the basis, and the basis change yields tableau $T^{(3)}$.

x_1	x_2	x_3	x_4	x_5	x_6	z			
0.4	0.2	-1.4	-0.2	1.0			=	0	
-7.8	-1.4	7.8	0.4		1.0		=	0	$T^{(1)}$
-2.3	-2.15	13.55	0.4			1.0	=	0	
1.0	0.5	-3.5	-0.5	2.5			=	0	
	2.5	-19.5	-3.5	19.5	1.0		=	0	$T^{(2)}$
	-1.0	5.5	-0.75	5.75		1.0	=	0	
1.0		0.4	0.2	-1.4	-0.2		=	0	
	1.0	-7.8	-1.4	7.8	0.4		=	0	$T^{(3)}$
		-2.3	-2.15	13.55	0.4	1.0	=	0	

Note that tableau $T^{(3)}$ is the same as tableau $T^{(1)}$ with the x variable columns shifted cyclically two columns to the right. It follows that this example will return to tableau $T^{(1)}$ after a further 4 iterations and therefore will cycle indefinitely with a cycle length of 6. In this example there are only two sets of coefficients: $T^{(3)}$ and $T^{(5)}$ are the same as $T^{(1)}$ with the x variable columns shifted cyclically 2 and 4 columns to the right, and $T^{(4)}$ and $T^{(6)}$ are the same as $T^{(2)}$ again shifted cyclically 2 and 4 columns to the right. We refer to such examples as 2/6-cycle examples. In this paper we restrict attention to 2/6-cycle examples as they are more elegant and easier to analyse than 6/6-cycle examples, such as Beale's example, which take 6 iterations to repeat the same coefficients. All the results here are demonstrated for 2/6-cycle examples. However the 2/6 property is not needed for the results and indeed 6/6-cycle examples can be formed by perturbing the 2/6-cycle examples given in this paper.

3 The form of 2/6-cycle examples

The following analysis was used to construct the above example. Let the 3×6 matrix $M^{(1)}$ be formed from the x columns of $T^{(1)}$ as follows

$$M^{(1)} = \left[\begin{array}{ccc} A & B & I \\ a & b & 0 \end{array}\right],$$

where A, B and I are 2×2 blocks of the constraint rows and a, b and 0 are 1×2 blocks of the objective row. To be able to pivot on the (1,1) and (2,2) entries in iterations 1 and 2, we require A to be non-singular. These pivoting operations yield tableau $T^{(3)}$, whose submatrix formed from the x columns has the form

$$M^{(3)} = \left[\begin{array}{ccc} I & A^{-1}B & A^{-1} \\ 0 & b-aA^{-1}B & -aA^{-1} \end{array}\right].$$

For the constraint pattern to repeat after these two iterations we require $A = A^{-1}B$ and $B = A^{-1}$, which occurs if and only if $A^3 = I$. This implies that the eigenvalues, λ, of A satisfy

$$\lambda^3 = 1 \iff (\lambda^2+\lambda+1)(\lambda-1) = 0. \tag{2}$$

For a 2×2 real matrix A there must either be 2 real eigenvalues or a complex conjugate pair.

It follows from (2) that if A has real eigenvalues they must both have the value 1, in which case the 2×2 matrix polynomial A^2+A+I has two real eigenvalues of 3 and is therefore non-singular. Since $(A-I)(A^2+A+I) = A^3 - I = 0$, it follows that $A = I$ in this case. It is then easy to show that $a = b = 0$, which is of no interest as it corresponds to a zero cost row.

The other possibility is that A has a complex conjugate pair of eigenvalues, and it follows from (2) that they must satisfy

$$\lambda^2+\lambda+1 = 0. \tag{3}$$

The characteristic equation of a general 2×2 matrix A is

$$\lambda^2 - (A_{11}+A_{22})\lambda + (A_{11}A_{22} - A_{21}A_{12}) = 0. \tag{4}$$

Equations (3) and (4) hold for the two distinct values of λ, so for a suitable 2/6-cycle example we require $A_{11}+A_{22} = -1$ and $A_{11}A_{22} - A_{21}A_{12} = 1$. From these it follows that

Table 1
Coefficient values over two iterations for 2/6-cycle examples

	x_1	x_2	x_3	x_4	x_5	x_6
	A_{11}	A_{12}	$-(A_{11}+1)$	$-A_{12}$	1	
$M^{(1)} =$	A_{21}	$-(A_{11}+1)$	$-A_{21}$	A_{11}		1
	-1	μ	$-(A_{11}+1)+\mu A_{21}$	$-A_{12}-\mu A_{11}$		
	1	$\frac{A_{12}}{A_{11}}$	$-(1+\frac{1}{A_{11}})$	$-\frac{A_{12}}{A_{11}}$	$\frac{1}{A_{11}}$	
$M^{(2)} =$		$\frac{1}{A_{11}}$	$\frac{A_{21}}{A_{11}}$	$-(1+\frac{1}{A_{11}})$	$-\frac{A_{21}}{A_{11}}$	1
		$\mu+\frac{A_{12}}{A_{11}}$	$\mu A_{21}-(2+A_{11}+\frac{1}{A_{11}})$	$-A_{12}(1+\frac{1}{A_{11}})-\mu A_{11}$	$\frac{1}{A_{11}}$	

$$-A_{21}A_{12} = 1 + A_{11} + A_{11}^2. \tag{5}$$

Conversely, any 2×2 matrix such that $A_{11}+A_{22} = -1$ and (5) holds has characteristic equation (3). Since a matrix satisfies its own characteristic equation, $A^2 + A + I = 0$, from which it follows that $A^3 = I$.

The objective function will repeat after 2 iterations if and only if $b - aA^{-1}B = a$ and $b = -aA^{-1}$. This occurs if and only if $a(A^2 + A + I) = 0$, which holds for all a since $A^2 + A + I = 0$. There is therefore no restriction on a. Since the scaling of the objective row is arbitrary we take a to have the form

$$a = [-1, \mu],$$

where there is no restriction on the value of μ. It follows that there is a three parameter family of 2/6-cycle examples: the parameters can be chosen as μ, A_{11} and A_{12}.

For arbitrary a, the vector b must satisfy

$$b = -aA^{-1}. \tag{6}$$

Since A is real and $A^3 = I$, $\det(A) = 1$. Hence

$$B = A^{-1} = \begin{bmatrix} -(A_{11}+1) & -A_{12} \\ -A_{21} & A_{11} \end{bmatrix},$$

$$\text{and} \quad b = [-(A_{11}+1) + \mu A_{21}, -A_{12} - \mu A_{11}],$$

and follows that the general form of $M^{(1)}$ and $M^{(2)}$ for the 2/6-cycle examples with the pivot sequence fixed is as in Table 1.

Proposition 1 summarises these results.

Proposition 1 *Assume the cost row is nonzero and the 2/6-cycle pattern of pivots is selected. Then the necessary and sufficient conditions for the coefficient pattern to repeat after two iterations are that the coefficients have the form given in tableau $M^{(1)}$ of Table 1, and that A_{11}, A_{21} and A_{12} satisfy (5).*

We now deduce the inequality relations that must be satisfied for the simplex method to select (1,1) and (2,2) as pivot elements. In order for (1,1) to be a pivot in tableau $M^{(1)}$ we require

$$A_{11} > 0. \tag{7}$$

From (5) and (7) it follows that A_{21} and A_{12} are nonzero and have opposite signs. If A_{21} is positive, A_{12} and hence $\frac{A_{12}}{A_{11}}$ are negative, so entry $M_{12}^{(2)}$ is negative and $M_{22}^{(2)}$ is positive, which is just the situation in the numerical example shifted cyclically one column to the right and with rows 1 and 2 interchanged. Hence without loss of generality we can take

$$A_{21} < 0, \tag{8}$$
$$A_{12} > 0. \tag{9}$$

It follows that the first row has the only positive entry in column 1 of $M^{(1)}$ and both constraint row entries in column 2 of $M^{(2)}$ are positive. Hence row 1 is the unique pivot candidate in iteration 1. There are two possible choices of pivot in column 2 of iteration 2. We shall use the *largest pivot* rule to break a tie. This rule chooses from the possible pivots the one of largest magnitude, and is the best choice from the point of view of numerical stability. To simplify the presentation we assume that if a tie remains after applying this rule, then the pivot in row 1 is chosen. This second tie-break rule therefore breaks the 2/6-cycle pattern if the pivot size criterion does not determine the pivot row. It follows that row 2 is the pivot choice in column 2 of iteration 2 if and only if

$$\frac{1}{A_{11}} > \frac{A_{12}}{A_{11}} \iff A_{12} < 1. \tag{10}$$

We have therefore proved the following proposition.

Proposition 2 *If the conditions of Proposition 1 are met and row selection ties are resolved by choosing the largest pivot and the columns are selected in the 2/6-cycle order, then the necessary and sufficient conditions for row 1 to be selected in odd iterations and row 2 in even iterations are $0 < A_{11}$ and $0 < A_{12} < 1$.*

The conditions guaranteeing that column 1 is chosen in $M^{(1)}$ by the most negative reduced cost rule rather than column 2 or 3 are

$$-1 < \mu, \tag{11}$$

$$-1 < -(A_{11}+1) + \mu A_{21} \iff \mu < \frac{A_{11}}{A_{21}}. \tag{12}$$

It follows from (7) and (8) that μ is negative. Column 1 is guaranteed to be chosen rather than column 4 if and only if

$$-1 < -A_{12} - \mu A_{11} \iff \mu < \frac{1-A_{12}}{A_{11}},$$

which is always true as this bound is positive by (7) and (10).

In $M^{(2)}$, column 5 has a positive cost entry so is not a candidate. The necessary and sufficient conditions for column 2 to be a candidate and be guaranteed to be chosen rather that columns 3 or 4 are

$$\mu < -\frac{A_{12}}{A_{11}}, \tag{13}$$

$$\mu < -\frac{(2 + A_{11} + \frac{1}{A_{11}} + \frac{A_{12}}{A_{11}})}{1 - A_{21}}, \tag{14}$$

$$\mu < -\frac{A_{12}(1 + \frac{2}{A_{11}})}{A_{11}+1}. \tag{15}$$

Comparing (13) and (15) we see that (13) is redundant if

$$\begin{aligned} & -\frac{A_{12}(1+\frac{2}{A_{11}})}{A_{11}+1} < -\frac{A_{12}}{A_{11}} \\ \iff & -A_{12}A_{11} - 2A_{12} < -A_{11}A_{12} - A_{12} \iff -A_{12} < 0, \end{aligned}$$

which is true by (9). Comparing (14) and (15), then using (8) and then (5), we see that (14) is redundant if

$$\begin{aligned} & -\frac{A_{12}(1+\frac{2}{A_{11}})}{A_{11}+1} < -\frac{(2 + A_{11} + \frac{1}{A_{11}} + \frac{A_{12}}{A_{11}})}{1 - A_{21}} \\ \iff & A_{12}(A_{11}+2)(1-A_{21}) > (A_{11}+1)(2A_{11} + A_{11}^2 + 1 + A_{12}) \\ \iff & A_{12}(A_{11} + 2 - A_{21}(A_{11}+2) - A_{11} - 1) > (A_{11}+1)^3 \\ \iff & -A_{12}A_{21}(A_{11}+2) + A_{12} > (A_{11}+1)^3 \\ \iff & (1 + A_{11} + A_{11}^2)(A_{11}+2) + A_{12} > (A_{11}+1)^3 \\ \iff & (A_{11}+1)^3 + 1 + A_{12} > (A_{11}+1)^3 \iff 1 + A_{12} > 0, \end{aligned}$$

which (9) shows is true. Comparing (12) and (13), then using (8) and then (5), we see that (12) is redundant if

$$-\frac{A_{12}}{A_{11}} < \frac{A_{11}}{A_{21}} \iff -A_{12}A_{21} > A_{11}^2 \iff A_{11}^2 + A_{11} + 1 > A_{11}^2,$$

which (7) shows is true.

We have now shown that (12), (13) and (14) are redundant, so (15) is always the tightest upper bound. From this and (11) it follows that μ must lie in the range

$$-1 < \mu < -\frac{A_{12}(A_{11}+2)}{A_{11}(A_{11}+1)}, \tag{16}$$

and there is a positive gap between these bounds if and only if

$$-1 < -\frac{A_{12}(A_{11}+2)}{A_{11}(A_{11}+1)} \quad \Longleftrightarrow \quad A_{12} < A_{11}\left(\frac{A_{11}+1}{A_{11}+2}\right). \tag{17}$$

If the left hand inequality in (16) is reversed, then column 2 will be chosen rather than column 1 in $M^{(1)}$, and if the right hand inequality is reversed, then column 4 will be chosen instead of column 2 in $M^{(2)}$. In either case the 2/6-cycle pattern will be broken. If either inequality in (16) holds as an equality, then the most negative reduced cost rule does not uniquely determine the column to enter the basis. To simplify presentation we assume that when this occurs a choice is made which breaks the 2/6-cycle pattern.

We have now proved the following proposition.

Proposition 3 *Assume that the the most negative reduced cost column selection rule and the largest pivot row degeneracy tie breaking rule are used. Then a 4 variable 2 constraint degenerate LP problem will have the 2/6-cycle pattern and cycle indefinitely if and only if the conditions of Propositions 1 and 2 hold and in addition (16) holds (which implies (17)).*

The unshaded area in Figure 1 (ignoring the dashed constraint) shows the region where the problem cycles indefinitely. Taking $A_{11} = 0.4$, $A_{12} = 0.2$ and $\mu = -2.15/2.3$ and then scaling the objective row by 2.3, produces the example given in Section 2.

A similar analysis to that leading to Proposition 1 for the case of a 2/4-cycle example shows that the cost row must be zero, so such examples cannot cycle. It is also straightforward to show that there can be no cycling examples with all pivots in the same constraint row, so there can be no problems with a single constraint. In the 2/6-cycle examples A_{12} and A_{21} must have different signs, so it follows from Table 1 that the even and odd iterations cannot be the same. Hence the 2/6-cycle examples are the simplest possible cycling examples.

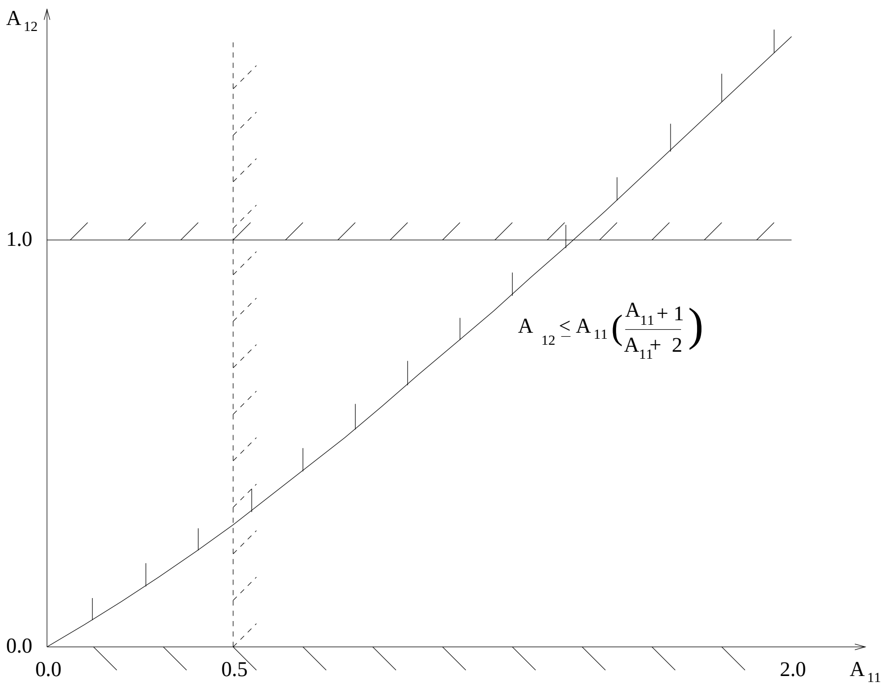

Fig. 1. Cycling region is unshaded. (Also cycles for EXPAND if $A_{11} \le \frac{1}{2}$)

4 A cycling steepest-edge example

In the previous sections the column was selected using the original Dantzig criterion of most negative reduced cost. In the steepest-edge method [8] the column is selected on the basis of the most negative ratio of the reduced cost to the length of the vector corresponding to a unit change in the nonbasic variable. This normally leads to a significant reduction in the number of iterations. When steepest-edge column selection is used on the example in Section 2, column 2 is chosen in $T^{(1)}$ instead of column 1 and in the following iteration the problem is shown to be unbounded so the simplex method terminates in 2 iterations. However by adding an extra row which affects the steepest-edge weights but not the choice of pivot row, one can construct a steepest-edge cycling example.

To preserve the 2/6-cycle pattern of the example, any extra constraints must behave like the objective row in that they must satisfy (6). We shall now construct an example that has a single candidate column in column 2 of $T^{(2)}$. We do this by selecting μ so that the x_4 objective coefficient in $T^{(2)}$ is zero. It follows from Table 1 that the required value is $\mu = -1.75$, and this results in the tableaux shown in Table 2, omitting the third rows. Note that column 1 would not now be selected in $T^{(1)}$ either by the most negative reduced cost criterion or by the steepest-edge criterion. We now introduce a constraint that will leave the steepest-edge weight of column 1 of $T^{(1)}$ unaltered but increase the weight of column 2. If the entries in this constraint are scaled up

Table 2
Cycling example with steepest-edge column selection

x_1	x_2	x_3	x_4	x_5	x_6	x_7	I			
0.4	0.2	-1.4	-0.2	1.0				=	0	
-7.8	-1.4	7.8	0.4		1.0			=	0	$T^{(1)}$
0.0	-20.0	156.0	8.0			1.0		=	1	
-1.0	-1.75	12.25	0.5				1.0	=	0	
1.0	0.5	-3.5	-0.5	2.5				=	0	
	2.5	-19.5	-3.5	19.5	1.0			=	0	$T^{(2)}$
	-20.0	156.0	8.0	0.0		1.0		=	1	
	-1.25	8.75	0.0	2.5			1.0	=	0	
1.0		0.4	0.2	-1.4	-0.2			=	0	
	1.0	-7.8	-1.4	7.8	0.4			=	0	$T^{(3)}$
		0.0	-20.0	156.0	8.0	1.0		=	1	
		-1.0	-1.75	12.25	0.5		1.0	=	0	

sufficiently, we can make steepest-edge choose column 1. Using $a = [0, -20.0]$ and applying (6) we get the third row of tableau $T^{(1)}$. We set the right-hand side of this constraint to 1, which ensures that this constraint is not involved in any of the pivot choices even when the matrix coefficients are perturbed by a small amount. With this extra row added the steepest-edge reduced costs for columns 1 and 2 of $T^{(1)}$ are -0.127 and -0.087, which leads to the selection of column 1 as required.

5 Analysis of the EXPAND procedure

The analysis given by Gill *et al* [7] of their EXPAND procedure proves that the objective function can never return to a value it had at a previous iteration. The EXPAND procedure however relaxes the constraints at each iteration, so the fact that the objective function continually improves does not prove that the method will not return to a previous basic solution. In Section 5.1 we describe the EXPAND procedure and in Section 5.2 derive the necessary and sufficient condition for cycling still to occur with the 2/6-cycle examples when using EXPAND. We do this by deriving an expression for the values of every variable at every iteration, a task that is made tractable by the special structure of the 2/6-cycle examples.

5.1 *The* EXPAND *ratio test*

The EXPAND approach to resolving degeneracy is described by Gill *et al* in [7] for the general bounded LP problem. The examples in this paper have single sided bounds and are of the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Mx = b, \quad x \geq 0. \end{array}$$

For simplicity, EXPAND is discussed here for this problem. Assuming that all the variables are feasible ($x \geq 0$), the standard ratio test for the simplex method determines the maximum step α in the direction p corresponding to the pivotal column such that the variables remain feasible, that is $x - \alpha p \geq 0$. For each j, the step which zeroes x_j is $\alpha_j = x_j/p_j$ if $p_j > 0$, otherwise $\alpha_j = \infty$. The maximum feasible step is therefore $\alpha = \alpha_r = \min_j \alpha_j$ and the variable to leave the basis is x_r.

EXPAND is based on the use of an increasing primal feasibility tolerance δ. During a particular 'current' simplex iteration, this tolerance has the value $\delta = \tilde{\delta} + \tau$, where $\tilde{\delta}$ was the value of δ in the previous iteration. At the beginning of the current iteration each variable satisfies its expanded bound $x_j \geq -\tilde{\delta}$. Since $-\delta < -\tilde{\delta}$, it is always possible to ensure that $\alpha > 0$, so there is a strict decrease in the objective function.

The EXPAND ratio test makes two passes through the entries in the pivotal column p.

- The first pass determines the maximum acceptable step $\alpha^{\max} > 0$ so that each basic variable satisfies its new expanded bound $x_j \geq -\delta$.
- The second pass determines a variable x_r to leave the basis. x_r is the variable with the largest acceptable pivot and is defined by
  $$r = \arg\max_j p_j \text{ such that } \alpha_j \leq \alpha^{\max} \text{ where } \alpha_j = \begin{cases} x_j/p_j & p_j > 0 \\ \alpha_j = \infty & \text{otherwise.} \end{cases}$$
  Define $\alpha^{\text{full}} = \alpha_r$. This is the step necessary to zero x_r. Note that if $x_r < 0$ and $p_r > 0$ then α^{full} will be negative.
- A minimum acceptable step
  $$\alpha^{\min} = \frac{\tau}{p_r}$$
  is calculated. If $x_r = -\tilde{\delta}$ then this is the maximum step that can be taken whilst maintaining feasibility with respect to the new expanded bounds.
- The actual step returned by the EXPAND ratio test is
  $$\alpha = \max(\alpha^{\min},\ \alpha^{\text{full}}).$$

We refer to these two alternative step sizes as the *min* and the *full* step.

The initial values of the nonbasic variables are zero. In the 2/6-cycle examples the initial values of the basic variables are also zero. The initial value of the expanding feasibility tolerance is denoted by τu, where $u \geq 0$, and the tolerance during iteration n is denoted by τu^n. It follows that $u^n = u + n$.

5.2 Conditions under which cycling occurs with the EXPAND *ratio test*

In this section we analyse the behaviour of the 2/6-cycle problems when using the EXPAND ratio test and derive necessary and sufficient conditions for the 2/6-cycle problems to cycle indefinitely.

The action of the EXPAND ratio test depends on whether the iteration number is even or odd, so we consider separately the behaviour in iterations $n = 2k+1$ and $n = 2k+2$ for $k \geq 0$. We assume that the pivot columns are selected in the 2/6-cycle order and derive necessary and sufficient conditions for EXPAND to select a pivot in the first row in odd iterations and have a unique pivot in the second row in even iterations. We also show that the min step is taken when the pivot is in row 1 and the full step is taken when the pivot is in row 2.

Let x_j^n denote the value of x_j at the start of iteration n. The subscripts of x are calculated modulo 6.

For iteration $2k+1$ the pivotal column is $[\,A_{11} \quad A_{21}\,]^T$ and the values of the basic variables at the start of the iteration are respectively x_{2k-1}^{2k+1} and x_{2k}^{2k+1}. Since $A_{21} < 0$ and $A_{11} > 0$, only x_{2k-1} moves towards its bound, so it is the sole candidate to leave the basis. The second pass of the EXPAND ratio test returns

$$\alpha^{\text{full}} = \frac{x_{2k-1}^{2k+1}}{A_{11}},$$

and if

$$x_{2k-1}^{2k+1} \leq \tau, \tag{18}$$

the min step will be taken so

$$\alpha = \frac{\tau}{A_{11}}.$$

It follows that if (18) holds, the changes in variable values are as given in row 1 of Table 3.

For iteration $2k+2$ the pivotal column is $[\, A_{12}/A_{11} \quad 1/A_{11} \,]^T$ and the values of the basic variables at the start of the iteration are respectively x_{2k+1}^{2k+2} and x_{2k}^{2k+2}. Since $A_{11} > 0$ and $A_{12} > 0$, both variables move towards their bound. The first pass of the EXPAND ratio test returns

$$\alpha^{\max} = \min\left(\frac{x_{2k+1}^{2k+2} + \tau u^{2k+2}}{A_{12}/A_{11}}, \frac{x_{2k}^{2k+2} + \tau u^{2k+2}}{1/A_{11}} \right).$$

A sufficient condition for the pivot to be in row 2 is that $A_{12} < 1$ and that the pivot is acceptable. It is acceptable if and only if

$$\begin{aligned} & \frac{x_{2k}^{2k+2}}{1/A_{11}} \le \alpha^{\max} \\ \iff \quad & A_{11} x_{2k}^{2k+2} \le A_{11} \min\left(\frac{x_{2k+1}^{2k+2} + \tau u^{2k+2}}{A_{12}}, x_{2k}^{2k+2} + \tau u^{2k+2} \right). \end{aligned}$$

Clearly $x_{2k}^{2k+2} < x_{2k}^{2k+1} + \tau u^{2k+2}$, so the pivot in row 2 is acceptable if and only if

$$A_{12} x_{2k}^{2k+2} \le x_{2k+1}^{2k+2} + \tau u^{2k+2}. \tag{19}$$

Also, provided that

$$x_{2k}^{2k+2} \ge \tau \tag{20}$$

then $\alpha^{\text{full}} = A_{11} x_{2k}^{2k+2} \ge \alpha^{\min} = A_{11}\tau$, so the full step α^{full} is taken and the EXPAND ratio test returns

$$\alpha = A_{11} x_{2k}^{2k+2}.$$

Hence if (19) and (20) hold, then the changes in values are as given in row 2 of Table 3.

From the changes in the values of variables given in Table 3, the expressions in Table 4 for the values of each variable over any two iterations are established by induction. To simplify notation we introduce the quantities s_k and S_k defined by

$$\begin{aligned} s_k &= \sum_{i=0}^{k} A_{11}^i, & S_k &= \sum_{i=0}^{k} (k+1-i) A_{11}^i, & &\text{for all } k \ge 0, \\ s_k &= 0, & S_k &= 0, & &\text{for all } k < 0. \end{aligned}$$

Note that since $A_{11} > 0$, s_k and S_k are nonnegative. Also

$$\begin{aligned} S_k - S_{k-1} &= s_k, \ \text{for all } k, \\ s_k &= 1 + A_{11} s_{k-1}, \ \text{for all } k \ge 0. \end{aligned} \tag{21}$$

Table 3. Changes in values of variables over two iterations

n	Entering	Leaving	Remaining	Step
$2k+1$	$x^{2k+2}_{2k+1} = x^{2k+1}_{2k+1} + \frac{\tau}{A_{11}}$	$x^{2k+2}_{2k-1} = x^{2k+1}_{2k-1} - \tau$	$x^{2k+2}_{2k} = x^{2k+1}_{2k} - \tau\frac{A_{21}}{A_{11}}$	Pivot row 1. Min step
$2k+2$	$x^{2k+3}_{2k+2} = x^{2k+2}_{2k}A_{11}$	$x^{2k+3}_{2k} = 0$	$x^{2k+3}_{2k+1} = x^{2k+2}_{2k+1} - x^{2k+2}_{2k}A_{12}$	Pivot row 2. Full step

Table 4. Expressions for the values of each variable over any two iterations. $s_k = \sum_{i=0}^{k} A_{11}^i, \ S_k = \sum_{i=0}^{k}(k+1-i)A_{11}^i.$

n	x^n_{2k+1}	x^n_{2k+2}	x^n_{2k+3}	x^n_{2k+4}	x^n_{2k+5}	x^n_{2k+6}	Expanded	Normal
	$-\tau S_{k-2}$	0	$-\tau S_{k-1}$	0	$\tau(1-S_k)$	$-\tau A_{21}s_{k-1}$		
$2k+1$	A_{11}	A_{12}	$-(A_{11}+1)$	$-A_{12}$	1	0	$\tau(1-S_k+u_{2k+1})\frac{1}{A_{11}}$	$\tau(1-S_k)\frac{1}{A_{11}}$
	A_{21}	$-(A_{11}+1)$	$-A_{21}$	A_{11}	0	1	∞	∞
	$\uparrow$							
	$\tau(\frac{1}{A_{11}} - S_{k-2})$	0	$-\tau S_{k-1}$	0	$-\tau S_k$	$-\tau\frac{A_{21}}{A_{11}}s_k$		
$2k+2$	1	$\frac{A_{12}}{A_{11}}$	$-(1+\frac{1}{A_{11}})$	$-\frac{A_{12}}{A_{11}}$	$\frac{1}{A_{11}}$	0	$\tau(\frac{1}{A_{11}} - S_{k-2} + u_{2k+2})\frac{A_{11}}{A_{12}}$	$\tau(\frac{1}{A_{11}} - S_{k-2})\frac{A_{11}}{A_{12}}$
	0	$\frac{1}{A_{11}}$	$\frac{A_{21}}{A_{11}}$	$-(1+\frac{1}{A_{11}})$	$-\frac{A_{21}}{A_{11}}$	1	$\tau(-\frac{A_{21}}{A_{11}}s_k + u_{2k+2})A_{11}$	$-\tau A_{21}s_k$
		$\uparrow$						
	$\tau(1-S_{k+1})$	$-\tau A_{21}s_k$	$-\tau S_{k-1}$	0	$-\tau S_k$	0		

The expressions in Table 4 allow condition (19) to be expressed as $G_k \geq 0$, where G_k for $k \geq 0$ is defined by

$$G_k = \frac{A_{12}A_{21}}{A_{11}} s_k + \frac{1}{A_{11}} - S_{k-2} + u^{2k+2}.$$

A necessary and sufficient condition on A_{11} for $G_k \geq 0$ is established by considering

$$\begin{aligned}\Delta G_k &\equiv G_{k+1} - G_k \\ &= \frac{A_{12}A_{21}}{A_{11}}(s_{k+1} - s_k) - (S_{k-1} - S_{k-2}) + u^{2k+4} - u^{2k+2} \\ &= -\frac{1 + A_{11} + A_{11}^2}{A_{11}} A_{11}^{k+1} - s_{k-1} + 2 \\ &= -(s_{k-1} + A_{11}^k + A_{11}^{k+1} + A_{11}^{k+2}) + 2 \\ &= -s_{k+2} + 2.\end{aligned}$$

It follows that $\Delta G_k \geq 0 \iff s_{k+2} \leq 2$. If $0 < A_{11} \leq \frac{1}{2}$, then s_{k+2} increases to a limit s_∞, where $s_\infty \leq 2$. In this case $\Delta G_k \geq 0$ for all k so $G_{k+1} \geq G_k$ for all k, and also $G_0 \geq u + \frac{1}{2} > 0$, so $G_k > 0$ for all $k \geq 0$. If $A_{11} > \frac{1}{2}$, then there exists an ϵ and K such that $s_{k+2} > 2 + \epsilon$ for all $k \geq K$. It follows that for suitably large k, $G_k < 0$. Hence for positive A_{11} the necessary and sufficient conditions for G_k to be nonnegative for all k is that $A_{11} \leq \frac{1}{2}$.

Proposition 4 *Assume that the conditions of Proposition 1 are met and the* EXPAND *row selection method is used and the columns are selected in the 2/6-cycle order. Then necessary and sufficient conditions for cycling to occur are that* $0 < A_{11} \leq \frac{1}{2}$ *and* $0 < A_{12} < 1$.

Proof:

Sufficient conditions:

We show by induction that the values of the variables at the start of odd iterations are as given in Table 4 and that these values lead to the correct choice of pivot row for the 2/6-cycle pattern.

Initially all the variables have the value zero, so $x_j^1 = 0$. Hence the values in Table 4 are correct for $n = 1$. Assume now that for some k the values in Table 4 are correct at the start of iteration $2k + 1$.

In iteration $2k + 1$, since s_k is non-negative, $x_{2k-1}^{2k} \leq \tau$, so (18) holds and it follows that the changes in the values of variables are as given by row 1 of Table 1. From this and (21) we get

$$x_{2k+6}^{2k+2} = -\tau A_{21} s_{k-1} - \tau \frac{A_{21}}{A_{11}} \tag{22}$$

$$
\begin{aligned}
&= -\tau\frac{A_{21}}{A_{11}}(A_{11}s_{k-1}+1)\\
&= -\tau\frac{A_{21}}{A_{11}}s_k.
\end{aligned}
$$

All the other values are straightforward, so we have deduced the values given in Table 4 at the start of iteration $2k+2$.

Substituting these values into (19) we see that the pivot in row 2 is acceptable if and only if

$$
-\frac{A_{12}A_{21}}{A_{11}}s_k \le \frac{1}{A_{11}} - S_{k-2} + u^{2k+2},
$$

which is true provided $A_{11} \le \frac{1}{2}$. Also

$$
x_{2k}^{2k+2} = -\tau\frac{A_{21}}{A_{11}}\sum_{i=0}^{k} A_{11}^i \ge -\tau\frac{A_{21}}{A_{11}} = -\tau\frac{1+A_{11}+A_{11}^2}{A_{12}} > \tau,
$$

since $A_{11} > 0$ (7) and $A_{12} < 1$. Hence (20) holds, so the changes in the values of variables are as given in row 2 of Table 3. The new value for x_{2k+1} is given by

$$
\begin{aligned}
x_{2k+1}^{2k+3} &= \tau\left(\frac{1}{A_{11}} - S_{k-2} + \frac{A_{12}A_{21}}{A_{11}}s_k\right)\\
&= \tau\left(\frac{1}{A_{11}} - S_{k-2} - \frac{1}{A_{11}}s_k - s_k - A_{11}s_k\right)\\
&= \tau\left(\frac{1}{A_{11}} - S_{k-2} - (\frac{1}{A_{11}} + s_{k-1}) - s_k - (s_{k+1}-1)\right)\\
&= \tau(1-S_{k+1}),
\end{aligned}
$$

which is the value given in Table 4. All the other values at the start of iteration $2k+3$ follow straightforwardly and are as shown in Table 4. These values are the values in Table 4 for k, with the k replaced by $k+1$. This completes the induction and shows that the 2/6-cycle pattern continues indefinitely.

Necessary conditions:

As discussed in Section 3, $A_{11} > 0$ and we can choose $A_{12} > 0$, in which case $A_{21} < 0$. Since $x_5^1 = 0$, the first iteration takes the min step and so $x_1^2 = \tau/A_{11}$. The pivot in row 1 in iteration 2 is acceptable if

$$
\begin{aligned}
& \frac{\tau}{A_{11}}\frac{A_{11}}{A_{12}} \le \alpha^{\max}\\
\iff\quad & \frac{\tau}{A_{12}} \le -\frac{\tau A_{21}}{A_{11}} + \tau(u+2)
\end{aligned}
$$

$$\iff \quad 1 \quad \leq \frac{1}{A_{11}} + 1 + A_{11} + u + 2, \tag{23}$$

which is true. Hence if $A_{12} > 1$, the pivot will be in row 1 in iteration 2 and the 2/6-cycle pattern will be broken. If $A_{11} > \frac{1}{2}$, then the argument prior to Proposition 4 shows there is a first value of k, $\hat{K}$ say, such that $G_{\hat{K}} < 0$. As shown above, for all $k < \hat{K}$ the 2/6-cycle pattern is maintained and the variable values are as in Table 4. Therefore in iteration $2\hat{K}$ the pivot in row 2 is not acceptable, so the pivot must be in row 1. This breaks the 2/6-cycle pattern. □

The conditions derived in Section 3 for the minimum reduced cost criterion to choose pivot columns in the 2/6-cycle pattern relied on the conditions $A_{11} > 0$ and $0 < A_{12} < 1$. These conditions have been established in Proposition 4 for the case of EXPAND row selection, so it follows that (16) and (17) still hold. From (17) and the fact that $A_{11} \leq \frac{1}{2}$ it follows that $A_{12} < \frac{3}{10}$, which is tighter that $A_{12} < 1$, which is therefore redundant. We have now shown the following proposition.

Proposition 5 *A 4 variable 2 constraint degenerate LP problem will have the 2/6-cycle pattern and cycle indefinitely when using the most negative reduced cost column selection rule and the* EXPAND *row selection rule if and only if the conditions of Proposition 1 hold and in addition* $0 < A_{11} \leq \frac{1}{2}$, $0 < A_{12}$ *and (16) holds (which implies relation (17)).*

The shaded area in Figure 1 including the $A_{11} \leq \frac{1}{2}$ constraint is the region where cycling occurs when using EXPAND. Note that the constraint $A_{12} < 1$ is now redundant. Also note that in the example (1), $A_{11} = 0.4 < 0.5$, so that example also cycles with EXPAND.

Finally note that the only way that EXPAND can escape from the 2/6-cycle pattern is for it to select the first row as pivot row in an even iteration, and if this occurs the resulting tableau has the form

x_1	x_2	x_3	x_4	x_5	x_6
$\frac{A_{12}}{A_{11}}$	1	$-\frac{A_{12}}{A_{11}} - \frac{1}{A_{12}}$	-1	$\frac{1}{A_{12}}$	0
$-\frac{1}{A_{12}}$	0	$\frac{A_{21}}{A_{11}} + \frac{1}{A_{12}} + \frac{1}{A_{11}A_{12}}$	-1	$-\frac{A_{21}}{A_{11}} - \frac{1}{A_{11}A_{12}}$	1
$-1 - \mu\frac{A_{11}}{A_{12}}$	0	$*$	$*$	$-\frac{\mu}{A_{12}}$	0

The constraint entries in the third and fourth columns are all negative and the objective function coefficients in all except these columns are nonnegative. Since the problem is unbounded we cannot be at an optimum, so one of these columns must be chosen. The next iteration will then produce an unbounded

step and the method will terminate.

The above results are independent of the EXPAND parameters u and τ. In [7] it is suggested that the initial tolerance τu be taken to be half of the feasibility tolerance δ_{f} to which the problem is to be solved. The value of τ is chosen so that after a large number of iterations (typically $K = 10000$) the expanded tolerance approaches δ_{f}, at which stage δ is reset to its original value δ_{i}. If this is done with the 2/6-cycle examples after an even iteration, then the problem returns to its initial state. If it is done after an odd iteration then it returns to the even iteration case but with the values all zero. It can be shown that in this case too the problem cycles, so that in neither case does resetting break the cycle pattern.

6 Conclusions

We have derived a three-parameter class of linear programming examples which cause the simplex method to cycle indefinitely. When written in standard form, these examples have two constraints and 6 variables and the coefficient pattern repeats every two iterations. These are the simplest possible examples for which the simplex algorithm cycles. We have derived 4 inequalities between the parameters and shown that these are the necessary and sufficient conditions for members of this class to cycle with Dantzig's form of the simplex method. We have shown how to extend the examples so that they also cycle when the steepest-edge column selection criterion is used. By adding the single bound, $A_{11} \leq \frac{1}{2}$, we were able to characterise the examples that also cycle using the EXPAND row-selection mechanism. This shows that despite the fact that in the EXPAND method the objective function is guaranteed to improve each iteration, the method is not guaranteed to prevent cycling. The cycling behaviour is independent of the EXPAND tolerance parameters. The bound $A_{11} \leq \frac{1}{2}$ is the only extra condition that had to be applied to ensure that an example which would cycle under the usual Dantzig rule, with largest pivot as the tie-breaker, would also cycle using EXPAND. This extra bound does reduce somewhat the number of cases that cycle, but does not eliminate the problem. Also the reduction is for problems where the degeneracy is exact. For problems which are close to degenerate EXPAND may cycle whereas the original simplex method in exact arithmetic will not.

All the coefficients in the examples (not just the 3 parameters) may be perturbed simultaneously by any small amount without destroying the cycling behaviour. The 2/6-cycle examples are therefore just points in a full dimensional set of counter-examples, so there is a positive probability of encountering cycling in randomly generated degenerate examples. In practice therefore the EXPAND procedure cannot be relied upon to prevent cycling. Provided we stay

within the class of the degenerate problems (*i.e.* keep the right-hand side 0) it is possible to vary the other coefficients by a significant amount. Indeed we have constructed examples where the values are totally different every 2 iterations and yet indefinite cycling still occurs with EXPAND.

The examples have been tested on our own implementation of EXPAND and using MINOS 5.4, which was written by the authors of EXPAND. In both cases if no preprocessing is done the examples cycle indefinitely. MINOS periodically does a reset operation (by default after 10000 iterations). This returns the problem to its initial state so cycling is still indefinite.

OSL [6] uses some techniques from EXPAND. In the examples in this paper, OSL 2.0 without scaling or preprocessing and with Dantzig pricing cycles for 30 iterations before reporting doing a perturbation, but it then continues to cycle indefinitely. CPLEX 4.0.7 without scaling or preprocessing cycles for 400 iterations before resolving the degeneracy by introducing a large perturbation. XPressMP 7.14 without scaling and with an even invert frequency cycles indefinitely. However the BQPD code of Fletcher [4] detects degeneracy at the start of the first iteration, changes to the dual, does one pivot, then finds that the dual is infeasible. This gives an improving direction in the primal, which resolves the degeneracy. Finally, BQPD detect unboundedness in this direction and terminates, having done one pivot in total.

References

[1] M. L. Balinski and R. E. Gomory. A mutual primal-dual simplex method. In R. L. Graves and P. Wolfe, editors, *Recent Advances in Mathematical programming*. McGraw-Hill, New York, 1963.

[2] E. M. L. Beale. Cycling in the dual simplex algorithm. *Naval Research Logistics Quarterly*, 2:269–75, 1955.

[3] R. Fletcher. Degeneracy in the presence of roundoff errors. *Linear Algebra and its Applications*, 106:149–183, 1988.

[4] R. Fletcher. Resolving degeneracy in quadratic programming. *Annals of Operations Research: degeneracy in optimization problem*, 47:307–334, 1993.

[5] R. Fletcher and J. A. J. Hall. Towards reliable linear programming. In G. A. Watson and D. F. Griffiths, editors, *Pitman Research Notes in Mathematics Series 228*, pages 89–104. Longman Scientific and Technical, 1990.

[6] J. J. H. Forrest and J. A. Tomlin. Implementing the simplex method for the Optimization Subroutine Library. *IBM Systems Journal*, 31(1):11–25, 1995.

[7] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. A practical anti-cycling procedure for linearly constrained optimization. *Math. Prog.*, 45:437–474, 1989.

[8] D. Goldfarb and J. K. Reid. A practical steepest-edge simplex algorithm. *Math. Prog.*, 12:361–371, 1977.

[9] P. M. J. Harris. Pivot selection methods of the Devex LP code. *Math. Prog.*, 5:1–28, 1973. [Reprinted in *Math. Prog. Study* 2:30–57, 1975.].

[10] P. Wolfe. A technique for resolving degeneracy in linear programming. *SIAM Journal Appl. Math.*, 11:205–211, 1963.